Question Answering:
CNLP at the TREC-2002 Question Answering Track

Anne R. Diekema, Jiangping Chen, Nancy McCracken, Necati Ercan Ozgencil, Mary D. Taffet, Ozgur Yilmazel, and Elizabeth D. Liddy

Center for Natural Language Processing
Syracuse University, School of Information Studies4-206 Center for Science and Technology
Syracuse, NY 1324-4100
www.cnlp.org
{diekemar,jchen06,njm,neozgenc,mdtaffet,oyilmaz,liddy}@syr.edu

Abstract
This paper describes the retrieval experiments for the main task and list task of the TREC-2002 question-answering track. The question answering system described automatically finds answers to questions in a large document collection. The system uses a two-stage retrieval approach to answer finding based on matching of named entities, linguistic patterns, keywords, and the use of a new inference module. In answering a question, the system carries out a detailed query analysis that produces a logical query representation, an indication of the question focus, and answer clue words.

1. Introduction
The Center for Natural Language Processing (CNLP) participated in the main task and the list task of the Question Answering track. The main task required answering 500 short fact-based questions, which have been extracted by NIST from MSNSearch and AskJeeves logs. Unlike previous years, the answer had to be exact, the answer string containing nothing but the answer itself. Also, unlike previous years, the answers to all 500 questions had to be ordered by answer confidence rather than by question number. This means that the answers that the system is most confident about should be ranked first, and the least confident should be ranked last. The scoring (see section 3) reflects a system's ability to determine how accurate a certain answer is. Not all questions had a known answer in the collection. Unanswerable questions have to be identified as such by the system to be counted correct.

The list task required answering 25 short fact-based list questions. List questions include an indication as to how many unique answer instances are needed to answer the question. A response to a list task question consisted of an unordered list of exact answers. The different answer instances could be found within single documents or across multiple documents, or a combination of both. Not all questions have all required answer instances in the collection.

This year there was a new document collection for the Question Answering track. Answers to both main task and list task questions had to be retrieved automatically from 1,033,461 documents from the following three sources: AP newswire, 1998-2000, New York Times newswire, 1998-2000, and Xinhua News Agency, 1996-2000.

2. System Overview
The CNLP question-answering system consists of four different processes: question processing, document processing, paragraph finding, and answer finding. The first three processes are similar to last year's system. [1] Changes were made to the answer finding module to adapt the system to the track's new requirements and to incorporate our new inference module.

2.1 Question processing

Question processing has two major parts - conversion of questions into a logical query representation and question focus recognition. Our L2L (Language-to-Logic) module was used this year to convert the query into a logical representation suitable for keyword matching and weighting in our answer finder module. Question focus recognition is performed in order to identify the type of answer expected, extraction of the number of answers required (used for the list task only), and assignment of a confidence level.

2.2 Document processing and paragraph finding

For document retrieval, we used the ranked document list as provided by NIST. The top 200 documents from the list for each question were extracted from the TREC collection as the source documents for paragraph finding. In the paragraph finding stage, we aim to select the most relevant paragraphs from the top 200 retrieved documents from the first stage retrieval step. Paragraph selection was based on keyword occurrences in the paragraphs. Paragraph detection is based on orthographic clues.

2.3 Answer finding

Four different strategies were applied to find the correct answers to questions. The four strategies were based on entity extraction, inference, answer patterns, and answer context, respectively. The latter three are still under development and did not contribute much to answer finding for TREC-2002. A triage program was developed to classify questions into different answer strategies based on their question type, question focus and the number of keywords.

2.3.1 Entity based approach

The entity-based strategy used the tagged paragraphs from the paragraph finding stage and identified different paragraph windows (different keyword combinations) within each paragraph. A weighting scheme was used to identify the most promising paragraph window for each paragraph. These paragraph windows were then used to find answer candidates based on the question focus. All answer candidates were weighted and the top one was selected. The strategy is similar to previous years with the addition of a new function for assigning an answer confidence score.

For each answer candidate, the system assigned a confidence score to indicate the systems' confidence regarding answer correctness. The confidence score was determined by the following factors: 1) number of keywords in the same sentence, 2) question focus, 3) categorization confidence, and 4) the presence of other answer candidates in the same sentence. A threshold was determined for the confidence judgment score. If a question had a top answer whose confidence score was below the threshold, the question would be marked as having no answer.

2.3.2 Inference based approach

The inference-based approach used the results of our existing event extraction system on text to assist in finding exact answers for queries that involve events, indicated by a verb. These extractions were saved as information frames, and we implemented an inference engine to search for extracted information and to use some simple forms of linguistic inference. While this question approach was designed to answer queries where identifying events was important to the answer, we also could utilize the inference engine to answer questions that needed two or more pieces of information to find the answer.

This approach starts with our existing generic entity and event extraction system. This system extracts event/agent/object information from sentences and also relation extraction about entities, primarily named entities, such as location, point-in-time and characteristic. The generic extraction is implemented using shallow parsing rules. To use generic extraction for Q&A, we processed the queries with the generic extraction system as well by adding shallow parsing rules for query forms and generating an answer template that represents the form of the answer as a generic extraction with the "exact answer" slot filled in with an unknown variable. Informally, in order to answer "When did Hawaii become a state?", we formulate a template "Hawaii became a state in time ?X", where ?X is a variable. For the "What <type of thing>" questions, we would generate a two-part answer template. For example, in "What king signed the Magna Carta?", we would generate both "?X signed the Magna Carta" and "?X is a king".

2.3.2.1 Rule patterns for queries

In order to analyze queries, we wrote shallow parsing rules that could recognize the query patterns. We describe a selection of those patterns here. Note that the query rules did not have to indicate additional qualifying phrases as those would be added by the generic extraction.

Patterns with possibly significant verb phrases:	
when do <nounphrase> <verbphrase> <nounphrase>	When did George Orwell write Animal Farm?
when do <nounphrase> <verbphrase>	When did Mt. St. Helens erupt?
Where do <nounphrase> <verbphrase>	Where did the ukulele originate?
who <verbphrase> <nounphrase>	Who invented baseball?
what do <nounphrase> <verbphrase>	What do bats eat?
Patterns with what (or which) <typeofthing>:	
what <typeofthing> do <nounphrase> <verbphrase>	What flower did Vincent van Gogh paint?
what <typeofthing> be <nounphrase> in	What hemisphere is the Philippines in?
what <typeofthing> be <nounphrase>	What color is a poison arrow frog?
what <typeofthing> be <nounphrase> <prepphrase>	What gasses are in the troposphere?
what <typeofthing> <verbphrase> <nounphrase>	What American composer wrote the music for "West Side Story"?
what <typeofthing> <verbphrase> <prepphrase>	What currency is used in Australia?

Table 1. Rule patterns for queries.

2.3.2.2 Query answer templates

When a query is processed by one of the query rules, one or more templates is generated to use in finding answers in the extraction database. An answer template is a frame in the same format as the frames in the extraction database. For each query, the frames are generated in the format of possible answers to the query, except that the unknown part is given as a variable, represented as the string ?X. (For TREC queries, we only needed to generate answer templates with one unknown variable.) It is the job of the inference engine to fill in a value for the variables, which will be an exact answer to the query.

In general, "when" queries ask for a property that is called "point-in-time" by the extraction system. For "do verb" forms, the first nounphrase in a sentence with an active verb is assumed to be the agent of that event.

when do <nounphrase> <verbphrase> <nounphrase> (When did George Orwell write Animal Farm?)

```
        event = do write
          agent  = George Orwell
          object  = Animal Farm
          point-in-time  = ?X
```

```
when do <nounphrase> <verbphrase>  (When did Mt. St. Helens erupt?)
        event = do erupt
          agent  = Mt. St. Helens
          point-in-time  = ?X
```

In a where query, there is a property "location" for events.

```
where do <nounphrase> <verbphrase>(Where did the ukulele originate?)
        event = do originate
          agent  = ukulele
          location  = ?X
```

Some query patterns are asking for the agent of the object of an event.

```
who <verbphrase> <nounphrase>(Who invented baseball?)
        event = invent
          agent  = ?X
          object  = baseball
```

```
what do <nounphrase> <verbphrase>  (What do bats eat?)
        event = eat
          agent = bats
          object = ?X
```

The "what <typeofthing>" patterns generate two frames for the two pieces of information. The second frame qualifies the answer as to what type of thing it is. The property is called "description" here, and there are several actual extraction properties that can be used to establish that the answer matches this description. Note that this frame is an entity frame.

```
what <typeofthing> do <nounphrase> <verbphrase>  (What flower did Vincent van Gogh paint?)
        event = paint
          agent = Vincent van Gogh
          object = ?X
        entity = ?X
          description = flower
```

The other "What <typeofthing>" query types similarly generate the second frame.

2.3.2.3 Extraction matching with the inference engine

For each query, the answer candidate documents were processed using the generic extraction system. The extractions were put into a database that we call the knowledge base. For answering queries, we then tried to match the template from the query, which is the "goal", with extractions in the knowledge base. We call the matcher the inference engine, but the types of inferencing that we are doing are linguistic in nature. We are not using inference rules that rely on world knowledge.

The inference engine tries to match all the frames representing the goal template. For each frame, it establishes that each attribute of the goal frame is present in the answer frame and that the values of each attribute "match". In order to match values of attributes, the inference engine has several rules to establish a match even if it is not an exact match.

Although we have not shown this in the examples so far, in addition to the string that is kept as the value of an attribute in the extraction frame, some string values also have links to an entity extraction frame. If such a link is present, the inference engine will also check that any additional attributes of that entity are also matched by the answer value. This is used in more complex queries that have additional modifiers.

Although the inference engine tries to match all of the attributes of the goal frame, it uses an abductive inference rule that allows a frame to match even when not all of the attributes are present, but with a lower probability of matching.

Finally, the inference engine has a set of axioms that embody linguistic knowledge about different forms of frames to try to match. If the goal frame has no match in the knowledge base, then these axioms are used to generate new goal frames that are sufficient to establish the answer.

An example of the types of linguistic alternatives is the changing of a goal event frame into an equivalent entity frame where that entity is described by the nominalization of the verb. This rule employs a list of such subject nominalizations.

event = invent	entity = ?X
agent = ?X	description = inventor
object = road traffic cone	modifier = of the road traffic cone

2.3.3 Pattern based approach

The pattern based approach is used for certain types of questions only: acronym, counterpart, definition, famous for, stand for, synonym, why. [1] We developed lexical pattern rules for answer extraction for these special question types. These patterns were used to identify text segments that could possibly provide an answer. Each of the answer identification patterns had its own confidence score indicating the likeliness of that pattern identifying for example, the meaning of a synonym. Unfortunately, the pattern-based approach did not prove effective for the TREC-2002 questions, partly because there were no definition questions this year. However, we find that the pattern based approach is useful in answering student's questions in the aerospace domain in a funded project for NASA.

2.3.4 Context-based approach

The context-based approach deals with those questions for which the system could not determine a question focus, which happens frequently (194 (39%) out of 500). When the system fails to identify a focus, the system attempts to find answers by using the context (the sentence in which the question keywords appear). This approach to answer finding is rather inexact and should be viewed as a last ditch effort.

3. Results

We submitted three runs for the TREC-2002 QA track: one run for the main task and two runs for the list task.

3.1 Main task results

Average over 500 questions	**SUT11IR1MT**
Confidence-weighted score	0.225
Number wrong	422
Number inexact	5
Number unsupported	9
Number right	64
Precision of recognizing no answer	0.167 (12 / 72)
Recall of recognizing no answer	0.261 (12 / 46)
Questions with rank above the median	47
Questions with rank on the median	427
Questions with rank below the median	26

Table 2. Question answering result for the main task.

The evaluation measure for the main task (see Table 2) is the confidence-weighted score (similar to the uninterpolated average precision measure from information retrieval). The score for an individual question is the number of correct answers up to and including that question divided by the number of questions answered so far. The score for the entire run is the mean of the individual questions' scores. The confidence-weighted score can range from 0 to 1 inclusive, with 1 a perfect score.

3.2 List task results

Average over 25 questions	**SUT11IR1LT**	**SUT11IR1LT2**
Average Accuracy	0.11	0.15
Questions with no answer found	17	15
Questions with rank above the median	5	8
Questions with rank on the median	17	15
Questions with rank below the median	3	2

Table 3. Question answering results for the list task.

The evaluation measure for the list task (see Table 3) is accuracy. The score for an individual question is the fraction of unique, correct instances over the target number of instances. The score for the entire run is the mean of the individual questions' scores. Accuracy can range from 0 to 1 inclusive, with 1 a perfect score.

4. Analysis

The analysis centers on the performance of our focus identification module and the contribution of each of the four different answer-finding approaches to question answering.

4.1 Main and list task performance

The large majority of the questions in the main task (84%) and list tasks (68%) were answered incorrectly. The number of questions for which our performance is the same as the median performance (of all participating systems) is close to (427 => 422), or identical (17 => 17, 15 => 15) to the number of questions that we answered incorrectly. These numbers seem to suggest that a lot of systems could not answer most of the questions. Further analysis is needed to determine why this is the case.

4.2 Focus identification

Focus identification is the most important procedure of query processing. It determines what answer strategy will be applied by the system to search for correct answers and also guides answer candidate selection. The question focus analysis is based on main task run (SUT11IR1MT).

The system correctly identified the focus for 301 questions out of 500 (60%), and incorrectly identified the focus for 5 questions (1%). There are 194 questions (39%) for which the system could not determine the focus (see Table 4). In cases where the focus is identified incorrectly no correct answers were found. When we look at the questions for which no focus could be determined at all we see that all the questions were answered incorrectly. These figures show that having a correct focus helps in finding a correct answer but definitely does not guarantee a correct answer.

500 questions	Correct question focus	Incorrect question focus	No determinable question focus
Correct answer	52 (10.4%)	0 (0%)	12 (2.4%)
Incorrect answer	249 (49.8)	5 (1%)	182 (36.4%)
Total	301 (60.2%)	5 (1%)	194 (38.8%)

Table 4. Question focus assignment.

4.3 Answer finding performance

The system applied several answer finding approaches this year, including a new inference module. However, the effort on these new approaches was limited due to the time constraints, leaving them in an earlier stage of development than we would have liked. When we look at the individual contributions of each of the modules (see Table 5) it becomes clear that the system still largely relies on the entity based approach for the identification of correct answers (49 out of 64 = 77%). Only 37 questions were sent to the inference engine, which managed to answer only 4 of them correctly. As pointed out previously, the pattern-based approach did not prove useful for TREC-2002 questions. Only one question was considered answerable by the pattern based approach and this question was answered incorrectly. The context approach, which is the module that handles questions for which there is no focus available, only answered 10 questions correctly out of the 163 that were assigned to this module. However, as a module that handles questions that no other module can handle, it still answered some questions that would otherwise have been lost. There were 8 questions for which no relevant paragraphs were found. These questions were deemed “unanswerable”. This proved correct for only one of them.

	Correct answers	Inexact answers	Unsupported answers	Wrong answers	Total
Entity based approach	49	5	9	218	281
Inference approach	4			33	37
Pattern based approach				1	1
Context based approach	10			163	173
No paragraphs found	1			7	8
Total	64	5	9	422	500

Table 5. Answer finding performance.

5. Conclusions and further research

It appears that most of the questions were not answered correctly by our system and that this is a common problem among participating systems. Analysis of the focus assignment module showed that having a correct focus helps in finding a correct answer but does not guarantee a correct answer. This suggests that improving the focus program to capture more question foci should not be the main center of our future research but rather we have to find other strategies to increase the number of questions we can answer correctly. Analysis of the four different answer finding approaches showed that the three modules other than the entity based module did not contribute much to finding correct answers. However, the reason for this could be that they are at an early stage in development. We will concentrate on further development of these modules.

References

[1] Chen, J., Diekema, A.R., Taffet, M.D., McCracken, N., Ozgencil, N. Ercan, Yilmazel, O., Liddy, E.D. (2002) *Question Answering : CNLP at the TREC-10 Question Answering Track*. In: The Tenth Text REtrieval Conference (TREC-2001). National Institute of Standards and Technology, Gaithersburg, MD., pp. 485-494.